CLAIMS

What is claimed is:

1. A system for offloading an input/output (I/O) task from a first computer to a second computer, comprising:

a client running on the first computer;

a server running on the second computer; and

at least one RDMA channel linking the first computer and the second computer, wherein the first computer and the second computer communicate in accordance with a protocol comprising a network discovery phase and an I/O processing phase.

2. The system of claim 1 wherein, in the I/O processing phase, read operations are implemented using RDMA and write operations are implemented using send operations.

3. The system of claim 1 wherein the protocol is used in association with a second network protocol.

4. The system of claim 3 wherein the second protocol is SMB.

5. The system of claim 3 wherein the second protocol is CIFS.

6. A computer-readable medium storing computer-executable instructions and computer-readable data comprising a computer program product for use in a system for

offloading an input/output (I/O) task from a first computer to a second computer, the system comprising:

at least one RDMA channel linking the first computer and the second computer, wherein the first computer and the second computer communicate in accordance with a protocol comprising a network discovery phase and an I/O processing phase.

7. A method for offloading an input/output (I/O) task from a first computer to a second computer, comprising:

discovering, by a client on the first computer and a server on the second computer, one or more shared RDMA-capable providers; and

posting, by the client, an I/O processing request for completion by the server on the second computer.

8. The method of claim 7 wherein the discovering one or more shared RDMA-capable providers further comprises:

obtaining, by the client, a server request resume key from the server;

opening, by the client, a pipe to the server;

sending, by the client over the pipe, a negotiate request; and

sending, by the server over the pipe, a negotiate response including a minimal list of common providers.

9. The method of claim 7, further comprising:

creating, by the client, an RDMA connection to the server over a shared RDMA-

capable provider; and

authenticating, by the client and the server, the RDMA connection.

10. The method of claim 9, further comprising:

registering, by the client, one or more files for use with the server over the RDMA connection.

11. The method of claim 10 wherein the registering one or more files comprises:

sending, by the client to the server, a register file message; and

sending, by the server to the client, a register file completion message.

12. The method of claim 9 wherein the authenticating the RDMA connection further comprises:

sending, by the client, an authenticate request message to the server, the authenticate request message including a key;

if the key matches a previous key sent by the server to the client, sending, by the server, an authenticate response message to the client.

13. The method of claim 12 wherein the previous key is a key contained in a negotiate response message sent by the server to the client.

14. The method of claim 12, further comprising:

sending, by the server to the client, a status response message to complete the

authenticating.

15. The method of claim 7 wherein the posting the I/O processing request comprises sending, by the client, one of (a) a close request, (b) a cancel request, (c) a read request, (d) a write request, (e) a vectored read request, and (f) a vectored write request.

16. The method of claim 15, further comprising:

completing, by the server, the read request and the vectored read request by sending data using RDMA write operations; and

completing, by the server, the write request and the vectored write request by sending data using normal send operations.

17. The method of claim 15 wherein the vectored write request includes a collapse flag in a header of the request.

18. The method of claim 7 wherein posting the I/O processing request further includes indicating whether the completion by the server should be in polling mode.

19. The method of claim 18 wherein the indicating whether the completion should be in polling mode comprises indicating that the completion should not be in polling mode by setting an interrupt flag in a header of the I/O processing request.

20. The method of claim 18, further comprising:

if the client indicates that the completion should not be in polling mode, completing, by the server, the I/O processing request by sending a status block to the first computer by way of RDMA transfer.

21. The method of claim 18, further comprising:

if the client indicates that the completion should be in polling mode, and the client has sent an interrupt request message to the server, sending, by the server to the client, an interrupt response message by way of an ordinary send.

22. The method of claim 7 wherein posting the I/O processing request further includes specifying a number of credits in a header of the request.

23. Computer-readable media storing computer-executable instructions for implementing a method for offloading an input/output (I/O) task from a first computer to a second computer, the method comprising:

discovering, by a client on the first computer and a server on the second computer, one or more shared RDMA-capable providers; and

posting, by the client, an I/O processing request for completion by the server on the second computer.

24. A method for managing buffers in an input/output offload protocol, comprising:

sending, by a server to a client, a delta credit message including an information field set to a number of credits, wherein, if the number is a negative number -N, the server requires

the client to retire N credits;

if the number of credits is a negative number -N, sending, by the client to the server, N credit messages, and otherwise sending, by the client to the server, one credit message; and

for each credit message sent by the client, sending, by the server to the client, a status response message.